МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ



**Дніпровський національний університет  
залізничного транспорту імені академіка В. Лазаряна**

Кафедра «Комп'ютерні інформаційні технології»

**Лабораторна робота №3**

**з дисципліни «Основи програмної інженерії»**

**на тему: «**Налагодження програм.**»**

Виконав:

студент гр.ПЗ1911

Сафонов Д.Є.

Прийняла:

Куроп'ятник О. С.

Дніпро, 2020

**Тема.** Налагодження програм.

**Мета.** Ознайомитися з методами та інструментами налагодження програм та отримати практичні навички їх використання.

# **Постановка задачі згідно загального та індивідуального завдання.**

Виконати розробку програми згідно індивідуального завдання та провести її налагодження. Процес налагодження документувати.

Вимоги до програми:

− матрицю представити у вигляді структури;

− створення матриці, її видалення, заповнення, виведення та обробки виконати в окремих функціях;

− глобальні змінні не допускаються;

− контроль значень вхідних даних (перевірка діапазону значень для чисел, коректна реакція на введення різних символів тощо).

Вимоги до тексту програми:

− коментарі щодо призначення кожної функції, її входу та виходу;

− самодокументованість коду: всі ідентифікатори повинні мати назви, що відповідають суті змінної або дії, яку вони позначають. Перевага надається англомовним назвам; − єдині стилі іменування змінних та функцій.

1. Дано символьну матрицю 10\*10, яка є полем гри «Морський бій». Згенерувати випадкове розташування кораблів, яке відповідає таким правилам:

− кораблі: 4 палуби – 1 шт., 3 палуби – 2 шт., 2 – 3 шт., 1 – 4 шт.;

− кораблі можуть розташовуватися лише по прямій;

− між кораблями має бути хоча б одна вільна клітинка;

− кораблі не можуть доторкатися один одного по діагоналі.

# **Протокол процесу налагодження програми.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № з/п | Опис помилки | Ситуація, коли проявляється | Вид помилки | Дії для локалізації | Можливі дії з усунення помилки | Застосовані дії усунення помилки |
| 1 | Помилка звернення за адресою | Звернення за межі масиву | Логічна | Помітив одразу | Перевірка значень | Перевірка значень |

# **Текст програми.**

“main.cpp”

#include <iostream>

#include "battleships.h"

int main() {

std::cout << "This application generates and prints board for the game \"battleships\"." << std::endl;

bool exit\_ = false;

do {

battleshipsBoard\* board1 = new battleshipsBoard;

board1->genBoard();

board1->printBoard();

delete board1;

char chExit;

std::cout << "Do you want to exit the programm(No means generate next board)?(Y/N): ";

std::cin >> chExit;

while ((chExit != 'Y' && chExit != 'y' && chExit != 'N' && chExit != 'n') || (std::cin.peek() != '\n')) {//until correct input

while (std::cin.get() != '\n');//clear istream

std::cout << "wrong input, try again." << std::endl;

std::cout << "Do you want to exit the programm?(Y/N): ";

std::cin >> chExit;

}

exit\_ = (chExit == 'Y' || chExit == 'y');

std::cin.get();//enter

system("cls");

} while (!exit\_);

system("pause");

return 0;

}

“battleships.h”

#ifndef \_\_BATTLESHIPS\_H\_\_

#define \_\_BATTLESHIPS\_H\_\_

#include <random>

#include <chrono>

#include <iostream>

#include <Windows.h>

enum seaField { empty, shipBody, bodyHit, miss, shipNeighbour };

enum direction { horizontal, vertical };

struct point {

int x;

int y;

};

struct battleshipsBoard {

battleshipsBoard();

~battleshipsBoard();

seaField\*\* board;

//checks if given ship can be placed in this object

bool validateShip(point start, direction dir, int length);

//places ship(use validation before!)

void placeShip(point start, direction dir, int length);

//generates board

void genBoard();

//generates, checks and places x-body ship in this object

void genXbody(int Xbodies);

//prints this object

void printBoard();

};

#endif

“battleships.cpp”

#include "battleships.h"

bool battleshipsBoard::validateShip(point start, direction dir, int length) {

bool usable = true;

if (dir == horizontal)

for (int i = 0; i < length && usable; i++)

if (start.y + i < 10)

usable = (this->board[start.x][start.y + i] == empty);

else

usable = false;

else//vertical

for (int i = 0; i < length && usable; i++)

if (start.x + i < 10)

usable = (this->board[start.x + i][start.y] == empty);

else

usable = false;

return usable;

}

void battleshipsBoard::placeShip(point start, direction dir, int length) {

if (dir == horizontal) {

for (int i = 0; i < length; i++)

this->board[start.x][start.y + i] = shipBody;

if (start.y - 1 >= 0)

this->board[start.x][start.y - 1] = shipNeighbour;

if (start.y + length < 10)

this->board[start.x][start.y + length] = shipNeighbour;

if (start.x - 1 >= 0) {

for (int i = 0; i < length; i++)

this->board[start.x - 1][start.y + i] = shipNeighbour;

if (start.y - 1 >= 0)

this->board[start.x - 1][start.y - 1] = shipNeighbour;

if (start.y + length < 10)

this->board[start.x - 1][start.y + length] = shipNeighbour;

}

if (start.x + 1 < 10) {

for (int i = 0; i < length; i++)

this->board[start.x + 1][start.y + i] = shipNeighbour;

if (start.y - 1 >= 0)

this->board[start.x + 1][start.y - 1] = shipNeighbour;

if (start.y + length < 10)

this->board[start.x + 1][start.y + length] = shipNeighbour;

}

}//

else {//vertical

for (int i = 0; i < length; i++)

this->board[start.x + i][start.y] = shipBody;

if (start.x - 1 >= 0)

this->board[start.x - 1][start.y] = shipNeighbour;

if (start.x + length < 10)

this->board[start.x + length][start.y] = shipNeighbour;

if (start.y - 1 >= 0) {

for (int i = 0; i < length; i++)

this->board[start.x + i][start.y - 1] = shipNeighbour;

if (start.x - 1 >= 0)

this->board[start.x - 1][start.y - 1] = shipNeighbour;

if (start.x + length < 10)

this->board[start.x + length][start.y - 1] = shipNeighbour;

}

if (start.y + 1 < 10) {

for (int i = 0; i < length; i++)

this->board[start.x + i][start.y + 1] = shipNeighbour;

if (start.x - 1 >= 0)

this->board[start.x - 1][start.y + 1] = shipNeighbour;

if (start.x + length < 10)

this->board[start.x + length][start.y + 1] = shipNeighbour;

}

}

}

void battleshipsBoard::genBoard() {

for (int i = 0; i < 10; i++)

for (int j = 0; j < 10; j++)

this->board[i][j] = empty;

genXbody(4);

for (int i = 0; i < 2; i++)

genXbody(3);

for (int i = 0; i < 3; i++)

genXbody(2);

for (int i = 0; i < 4; i++)

genXbody(1);

}

void battleshipsBoard::genXbody(int Xbodies) {//Xbodies supposed to be 1-4

std::uniform\_int\_distribution<int> XYdistribution(0, 9);

std::uniform\_int\_distribution<int> directioDistribution(0, 1);

std::default\_random\_engine generator(time(NULL));

point start;

direction dir;

bool gen = true;

do {

start.x = XYdistribution(generator);

start.y = XYdistribution(generator);

dir = static\_cast<direction>(directioDistribution(generator));

gen = validateShip(start, dir, Xbodies);

} while (!gen);

placeShip(start, dir, Xbodies);

}

void battleshipsBoard::printBoard() {

HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);

std::cout << " A B C D E F G H I J" << std::endl;

for (int i = 0; i < 10; i++) {

std::cout << i;

for (int j = 0; j < 10; j++)

if (this->board[i][j] == shipBody) {

SetConsoleTextAttribute(hConsole, 127);

std::cout << "[]";

SetConsoleTextAttribute(hConsole, 15);

}

else

std::cout << "[]";

std::cout << std::endl;

}

}

battleshipsBoard::battleshipsBoard() {

this->board = new seaField \* [10];

for (int i = 0; i < 10; i++)

board[i] = new seaField[10];

}

battleshipsBoard::~battleshipsBoard() {

for (int i = 0; i < 10; i++)

delete[]board[i];

delete[]board;

}

# **Аналіз результатів та висновки щодо зручності та ефективності застосованих методів та використаних інструментів.**

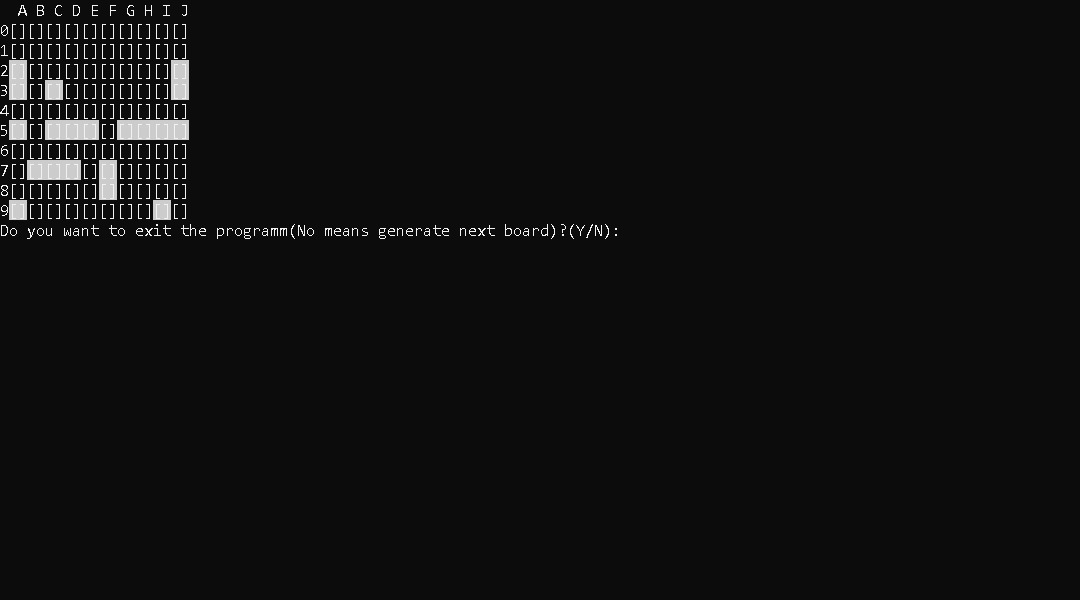


Рисунок 1

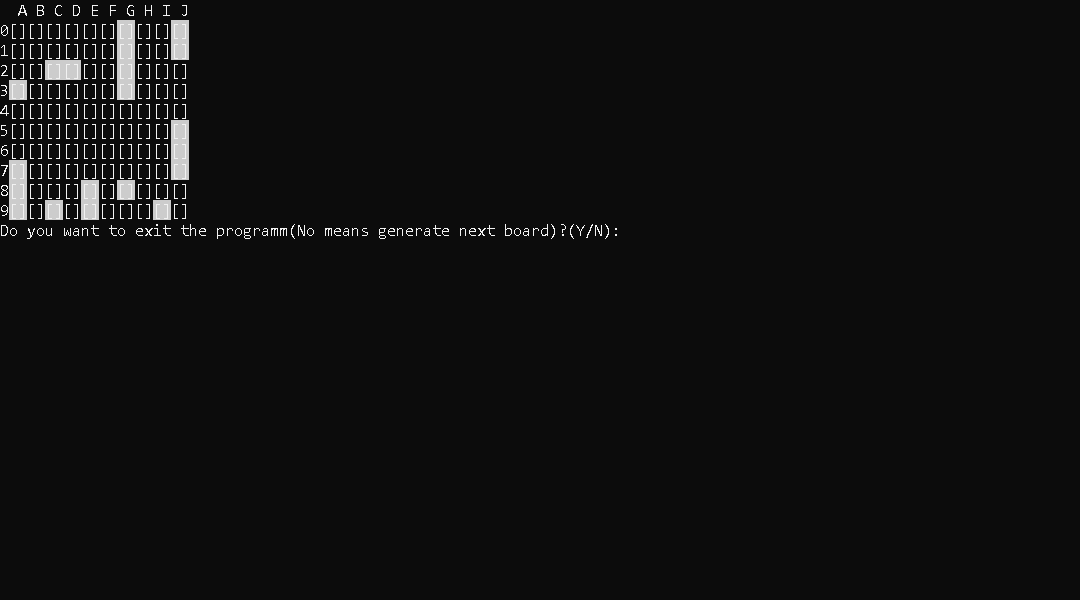


Рисунок 2

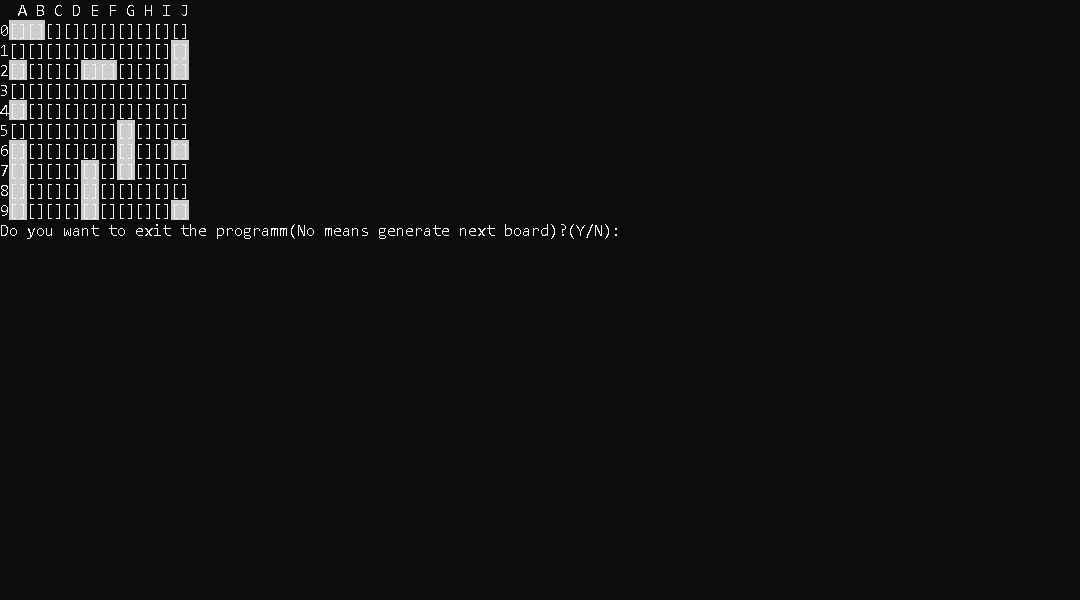


Рисунок 3

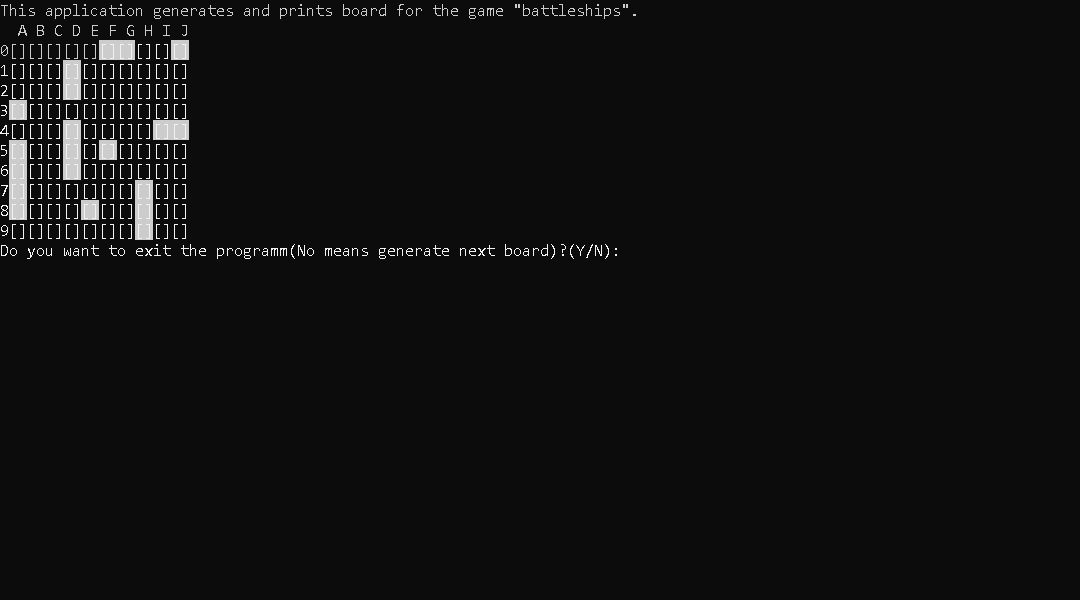


Рисунок 4

На мою думку найкращій метод запобігання помилок – попереднє проектування програми. Якщо добре спроектувати програму, то можна уникнути більшість складних помилок, і мати діло лише з простими помилками, які можна помітити без додаткового налагодження.

Якщо помилку не видно одразу, скоріш за все можна буде її локалізувати за допомогою контрольного друку. Якщо і це не вдалося скоріш за все помилка логічна і дуже велика, і потребує точного аналізу коду.